UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/883,508	06/19/2001	Jeffrey A. Bedell	53470.003042	8696

21967 7590 01/16/2009

HUNTON & WILLIAMS LLP
INTELLECTUAL PROPERTY DEPARTMENT
1900 K STREET, N.W.
SUITE 1200
WASHINGTON, DC 20006-1109

EXAMINER	
ZHEN, LI B	

ART UNIT	PAPER NUMBER
2194	

MAIL DATE	DELIVERY MODE
01/16/2009	PAPER

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

UNITED STATES PATENT AND TRADEMARK OFFICE

BEFORE THE BOARD OF PATENT APPEALS
AND INTERFERENCES

Ex parte JEFFREY A. BEDELL, MICHAEL CODINI, ARTURO GAY, WILLIAM HURWOOD, BENJAMIN Z. LI, FABRICE MARTIN, RAMKUMAR RAMACHANDRAN, STEPHEN S. TRUNDLE, ABHIMANYU WARIKOO, and KYLE N. YOST

Appeal 2008-3433
Application 09/883,508
Technology Center 2100

Decided: January 15, 2009

Before JAMES D. THOMAS, JEAN R. HOMERE, and STEPHEN C. SIU, *Administrative Patent Judges*.

SIU, *Administrative Patent Judge*.

STATEMENT OF THE CASE

This is a decision on appeal under 35 U.S.C. § 134(a) from the Examiner's rejection of claims 1-18. We have jurisdiction under 35 U.S.C. § 6(b). An Oral Hearing was held on November 18, 2008. We affirm-in-part.

The Invention

The disclosed invention relates generally to manipulating program objects (Spec. 1). Specifically, a system checks dependencies between objects as they are moved from one stage to another (Spec. 2).

Independent claim 1 is illustrative:

> 1. A computer implemented method for managing groups of objects for use in a reporting system project comprising the steps of:
>
> receiving a command to perform a selected function on a selected object;
> automatically identifying dependent objects referred to by the selected object;
> determining using a computer processor an appropriate manner of executing the selected function on the selected object;
> determining using a computer processor appropriate functions to be performed on the dependent objects;
> automatically causing the appropriate functions to be performed on the dependent objects; and
> automatically causing the execution of the selected function on the selected object in the appropriate manner.

The References

The Examiner relies upon the following references as evidence in support of the rejections:

Mariani	US 5,854,932	Dec. 29, 1998

Almond	US 6,112,024	Aug. 29, 2000
Fontana	US 6,167,563	Dec. 26, 2000

The Rejections

1. The Examiner rejects claims 1, 3-5, 10-12, and 16-18 under 35 U.S.C. § 103(a) as being unpatentable over Mariani and Fontana.
2. The Examiner rejects claims 2, 6-9, and 13-15 under 35 U.S.C. § 103(a) as being unpatentable over Mariani, Fontana, and Almond.

ISSUES

Issue #1(claims 1, 3, 18)

The Examiner finds that Mariani discloses that "the recompiling function [i.e., the "selected function"] determines which files to recompile" (Ans. 15) and that "[b]y determining the changed files and its dependent files, the recompile function determines a particular manner of rebuilding of the executable program" (Ans. 16).

Appellants assert that the Examiner "continues to rely on the simple determination of whether or not to recompile a file" and that "[a]voiding recompiling is . . . not 'an appropriate manner of executing the selected function on the selected object'" (Reply Br. 5).

Did Appellants demonstrate that the Examiner erred in finding that Mariani discloses or suggests determining a manner of executing a selected function on a selected object?

Issue #2

Appellants assert that "updating dependent components in response to a user prompt does not satisfy the 'automatically identifying dependent objects,' 'automatically causing the appropriate functions to be performed on the dependent objects' and 'automatically causing the execution of the selected function on the selected object in the appropriate manner' recitations" (Reply Br. 6-7).

The Examiner finds that "[a]utomatically performing a function is interpreted as performing the function with a machine" (Ans. 18) and that "Fontana's invention . . . [is] automatically performed by the machine that is executing the program code" (*id.*).

Did Appellants demonstrate that the Examiner erred in finding that Fontana discloses automatically identifying dependent objects, causing functions to be performed, and executing a selected function on a selected object in an appropriate manner?

Issue #3

Appellants assert that "the Office has failed to provide any proper motivation for modifying Mariani [with Fontana], so the proposed modification fails" (App. Br. 12).

Did Appellants demonstrate that the Examiner erred in combining Mariani with Fontana?

Issue #4

Appellants assert that "[t]here is no disclosure of 'managing objects with and between projects of a reporting system' in Mariani" (Reply Br. 7).

The Examiner finds that "the recitation 'managing objects with and between projects of a reporting system,' has not been given patentable weight because the recitation occurs in the preamble" (Ans. 19).

Did Appellants demonstrate that the Examiner erred in determining that the recitation of "a reporting system" is given no patentable weight?

Issue #5

Appellants assert that "Mariani . . . does not disclose a method or system wherein . . . each object has a unique identifier and a version identifier" (Reply Br. 8) or "manipulating objects between projects" (*id.*).

The Examiner finds that Mariani discloses that "each object has a unique identifier [search for a name in the scope of the class; col. 13, lines 25-50] and a version identifier [date and time of the last change made to the header files; col. 10, lines 8-45]" (App. Br. 7). The Examiner further finds that Mariani discloses "manipulating objects within and between projects [For use in creating and editing the source code files 60 and header files 62 of the user's project . . .] (App. Br. 7).

Did Appellants demonstrate that the Examiner erred in finding that the Mariani reference discloses that each object has a unique identifier and a version identifier and manipulating objects between projects?

Issue #6

Appellants assert that Mariani fails to disclose "a system 'wherein the operational module interfaces with projects that reside in various environments,'" (Reply Br. 8).

The Examiner finds that "Mariani discloses a development environment that provides access to third party code libraries [i.e., Microsoft Foundation Classes, or Object Linking and Embedding; col. 8, lines 22-43]" (Ans. 20).

Did Appellants demonstrate that the Examiner erred in finding that the Mariani reference discloses projects that reside in various environments?

Issue #7

The Examiner finds that Mariani discloses that "the minimal rebuild system compiles the source file . . . [and] makes a copy of the current object file" (Ans. 21).

Appellants assert that a "copy made as part of a system process to recompile cannot be logically interpreted as disclosing a copy made in

response to 'receiving a command to copy a selected object from a source project to a destination project.'" (Reply Br. 9).

Did Appellants demonstrate that the Examiner erred in finding that the Mariani reference discloses receiving a command to copy a selected object from a source project to a destination project?

Issue #8

Examiner finds that Almond discloses "an object contained in metadata [maps an object into a meta model which facilitates version control; col. 3, lines 1-54 and col. 6, lines 40-67] of an on-line analytical processing program [View reports--quickly view project activity status; col. 39, lines 15-39].

Appellants assert that "the terms meta model and metadata are not the same" (Reply Br. 9).

Did Appellants demonstrate that the Examiner erred in determining that the cited references disclose or suggest metadata?

FINDINGS OF FACT

The following Findings of Facts (FF) are shown by a preponderance of the evidence.

1. Mariani discloses that "the minimal rebuild system **100** determines when recompiling can be avoided by determining how the object

code files **82** are dependent on the header files **62** ('dependency analysis' **112**)" (col. 9, ll. 5-8).

2. Mariani discloses that "the minimal rebuild system **102** recompiles all object code files **82** that are dependent on header files **62** detected as having been modified" (col. 10, ll. 39-41) and "combines the change and dependency information . . . to determine which of the object code files **82** are to be recompiled (and conversely, which recompilations to omit) when the user's project is rebuilt" (col. 12, ll. 1-6).
3. Mariani discloses that "the user can directly modify or add C++ statements to the source code files **60** and header files **62**" (col. 8, ll. 11-13).
4. Mariani discloses that "the dependency types of an object code file on a class that are tracked by the minimal rebuild system **100** include dependencies on a search for a name in the scope of the class" (col. 13, ll. 25-28).
5. Mariani discloses "various tools **54** for creating, editing, and compiling source code files **60** and header files **62** for a user's programming project which is to be implemented in an executable program **64**" (col. 7, ll. 62-65).

6. Mariani discloses that a "change detection in the class wizard **110** involves simply notifying the minimal rebuild engine **102** of the particular change that the user selects to have made" (col. 10, ll. 2-5).
7. Mariani discloses that "for the header files **62** that are modified outside the class wizard, the minimal rebuild system **100** avoids no more recompiles than the prior make utilities" (col. 10, ll. 42-45).
8. Mariani discloses that "the minimal rebuild engine **102** is informed of changes to class declarations in the header files **62** by the compiler **80**" (col. 10, ll. 46-48).
9. Mariani discloses that the "compiler driver **104** controls compiling of the object code files **82** by the compiler **80**" (col. 12, ll. 18-19) and "[f]or object code files **82** that the minimal rebuild engine **102** determines recompilation can be avoided, the compiler driver **104** causes updating of such object code files' browser and debugger information" (col. 12, ll. 23-26).
10. Mariani discloses "[i]f the changes made to a header file intersect the dependencies of an object code file on the header file (meaning that the change could affect the object code file), then the object code file is recompiled. If there is no intersection of the changes with the object code file's dependencies on its header files (indicating the change cannot affect the object code file), then the object code file is not recompiled" (col. 12, ll. 9-15).

11. Fontana discloses "dependency objects are obtained from the repository . . . [and] are then analyzed to identify dependent components" (col. 7, ll. 24-26).
12. Fontana discloses that "the dependent components are accordingly updated (block 76)" (col. 7, ll. 52-53).
13. Fontana discloses "the components built in block 74 and the external components are put back into the repository (block 78)" (col. 7, ll. 55-56).
14. Almond discloses a "meta model **300**" that "allows the system to map objects to representations other than that provided in relational databases" and "serves as a container which facilitates version control" (col. 6, ll. 42-46).

PRINCIPLES OF LAW

35 U.S.C. § 103(a)

> Section 103 forbids issuance of a patent when "the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains."

KSR Int'l Co. v. Teleflex Inc., 127 S. Ct. 1727, 1734 (2007).

"What matters is the objective reach of the claim. If the claim extends to what is obvious, it is invalid under *§ 103*." *KSR Int'l Co. v.*

Teleflex, Inc., 127 S. Ct. at 1742 (2007). In *KSR*, the Supreme Court emphasized "the need for caution in granting a patent based on the combination of elements found in the prior art," *Id.* at 1739, and discussed circumstances in which a patent might be determined to be obvious. *KSR,* 127 S. Ct. at 1739 (citing *Graham v. John Deere Co.*, 383 U.S. 1, 12 (1966)). The Court reaffirmed principles based on its precedent that "[t]he combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable results." *Id.* The operative question in this "functional approach" is thus "whether the improvement is more than the predictable use of prior art elements according to their established functions." *Id.* at 1740.

The Federal Circuit recently recognized that "[a]n obviousness determination is not the result of a rigid formula disassociated from the consideration of the facts of a case. Indeed, the common sense of those skilled in the art demonstrates why some combinations would have been obvious where others would not." *Leapfrog Enters., Inc. v. Fisher-Price, Inc.*, 485 F.3d 1157, 1161 (Fed. Cir. 2007) (citing *KSR*, 127 S. Ct. 1727, 1739 (2007)). The Federal Circuit relied in part on the fact that Leapfrog had presented no evidence that the inclusion of a reader in the combined device was "uniquely challenging or difficult for one of ordinary skill in the art" or "represented an unobvious step over the prior art." *Id.* at 1162 (citing *KSR*, 127 S. Ct. at 1740-41).

ANALYSIS

Issue #1

Did Appellants demonstrate that the Examiner erred in finding that Mariani discloses or suggests determining a manner of executing a selected function on a selected object?

Mariani discloses implementing a programming project in an executable program (FF 5). Using a broad but reasonable interpretation of "determining a manner of executing" to include determining a possible, customary, or preferred way of doing something (i.e., executing the executable program), we agree with the Examiner that Mariani discloses determining a manner of executing the program or function. For example, one "manner" (i.e., way of accomplishing a task) in which Mariani executes the program or function includes executing the program while "avoiding recompiling the source code files 60 . . . and the header files 62 . . . into the object code files 82 . . . where any changes to the header files 62 do not affect the resulting object code files 82" (col. 9, ll. 1-5). In another "manner" of executing a program, Mariani discloses using "header files 62 that are modified outside the class wizard" (col. 10, ll. 42-43) such that "the minimal rebuild system **100** avoids no more recompiles than the prior make utilities" (col. 10, ll. 43-44). In either case, Mariani discloses determining an "appropriate manner" of executing the function.

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claims 1-18 with respect to issue #1.

Issue #2

Did Appellants demonstrate that the Examiner erred in finding that Fontana discloses automatically identifying dependent objects, causing functions to be performed, and executing a selected function on a selected object in an appropriate manner?

Fontana discloses obtaining dependency objects, analyzing the objects to identify dependent components, updating dependent components, building components, and putting external components back into a repository (FF 11-13). We agree with the Examiner that Fontana discloses identifying dependent objects (e.g., identifying dependent components), performing functions on the dependent objects (e.g., updating dependent components), and causing the execution of a function on an object in an appropriate manner. Appellants argue that "updating dependent components in response to a user prompt does not satisfy the 'automatically identifying dependent objects,' 'automatically causing the appropriate functions to be performed on the dependent objects' and 'automatically causing the execution of the selected function on the selected object in the appropriate manner' recitations" (Reply Br. 6-7) but do not cite specific structural or functional

differences between Fontana's disclosure and the disputed features of claim 1.

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claims 1-18 with respect to issue #2.

Issue #3

Did Appellants demonstrate that the Examiner erred in combining Mariani with Fontana?

Mariani discloses known elements of identifying modified files that are dependent on object code files and compiling the object code files in a project (FF 2). Fontana discloses identifying dependency objects and components (FF 11) and updating or modifying the components (FF 12). Hence, both Mariani and Fontana disclose known functions of identifying components or objects and their dependencies and modifying or updating the components or objects to achieve the expected results of modifying or updating objects or components and/or their dependent objects or components. Appellants have not demonstrated that the combination of Mariani and Fontana would have resulted in anything more than what one of ordinary skill in the art would have expected – namely, performing functions (e.g., updating, modifying or compiling) on objects or components.

> The combination of familiar elements according to known methods is likely to be obvious when it does no more than yield predictable

> results. . . . [W]hen a patent 'simply arranges old elements with each performing the same function it had been known to perform' and yields no more than one would expect from such an arrangement, the combination is obvious.

KSR at 1395-66 (citing *Sakraida v. AG Pro, Inc.*, 425 U.S. 273, 282 (1976)).

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claims 1-18 with respect to issue #3.

Issue #4

Did Appellants demonstrate that the Examiner erred in determining that the recitation of "a reporting system" is given no patentable weight?

We consider the Examiner's rejection of claims 10, 12, 16, and 17 as being unpatentable over Mariani and Fontana. Since Appellants' arguments have treated these claims as a single group which stand or fall together, we select independent claim 10 as the representative claim for this group. *See* 37 C.F.R. § 41.37(c)(1)(vii).

"The preamble of a claim does not limit the scope of the claim when it merely states a purpose or intended use of the invention." *In re Paulsen*, 30 F.3d 1475, 1479 (Fed. Cir. 1994). Claim 10 recites a system for managing objects between projects of a reporting system. The body of claim 10 recites receiving a user command and a module interfacing with projects,

determining an appropriate manner of executing a user command, determining functions to be performed, and executing a user command in an appropriate manner. Thus, while the preamble of claim 10 recites an intended purpose of a system application as managing objects in projects of "a reporting system," claim 10 does not recite within the body of the claim any features that are specific to the "reporting system."

Because Appellants have not demonstrated why the preamble phrase of "projects of a reporting system" must be considered a limitation, we are not persuaded of error in the rejection of claim 10.

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claim 10, or of claims 12, 16, and 17, which fall therewith with respect to issue #4.

Issue #5

Did Appellants demonstrate that the Examiner erred in finding that the Mariani reference discloses that each object has a unique identifier and a version identifier and manipulating objects between projects?

We consider the Examiner's rejection of claims 4 and 5 as being unpatentable over Mariani and Fontana. Since Appellants' arguments have treated these claims as a single group which stand or fall together, we select

independent claim 4 as the representative claim for this group. *See* 37 C.F.R. § 41.37(c)(1)(vii).

Appellants assert that a "date time stamp . . . does not disclose a version identifier" (Reply Br. 8). However, Mariani discloses that an engine "tracks (i.e., stores) the date and time of the last change made to the header files" (col. 10, ll. 17-18) and identifies changes in different versions of a header file by comparing "the header file's current date stamp (which indicates the date and time it was last saved) to the previously stored date stamp of the last class wizard edit to that header file" (col. 10, ll. 23-26). Hence, Mariani discloses date/time stamps associated with different versions (i.e., modified versions) of header files. Construing the term "version identifier" broadly but reasonably to include any name or symbol ("identifier") that indicates or names a form or variety of a component ("version"), we agree with the Examiner that the date/time stamp of Mariani that identifies an edited or modified form of a header file includes a "version identifier" as recited in claim 4.

In addition, Mariani discloses that "header files . . . of a project are compiled into one or more object code files (generally having file names with a '.obj' extension)" (col. 2, ll. 19-21). Thus, Mariani also discloses that object code files have "file names." We find that the "file names" that identify specific object code files that contain compiled header files

constitute "a unique identifier" as recited in claim 4 because the file names uniquely identify the object code files.

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claim 4, and claim 5, which falls therewith with respect to issue #5.

Issue #6

Did Appellant demonstrate that the Examiner erred in finding that the Mariani reference discloses projects that reside in various environments?

Mariani discloses a "development environment **52**" that "provides a code library **76**" (col. 8, ll. 23-24). The development environment of Mariani includes, for example, "one or more editors **70**, and automated source code generators **72**" (col. 8, ll. 9-10). Hence, the development environment of Mariani includes any set of the "one or more editors" or "automated source code generators" to create a particular set of conditions for affecting an activity (i.e., compiling object code files). Because the set of conditions created in the development environment may be of different kinds depending on the components used, we agree with the Examiner that Mariani discloses managing projects in various (i.e., different kinds of) environments (i.e., set of conditions for compiling object code files).

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claim 11 with respect to issue #6.

Issue #7

Did Appellants demonstrate that the Examiner erred in finding that the Mariani reference discloses receiving a command to copy a selected object from a source project to a destination project?

The Examiner finds that Mariani discloses that a minimal rebuild system "makes a copy of the current object file" (Ans. 21). While we agree with the Examiner that Mariani discloses "the minimal rebuild system 102 recompiles all object code files **82** . . ." (col. 10, ll. 39-40), we do not identify a teaching or suggestion in Mariani that the minimal rebuild system also copies a selected object from a source project to a destination project. While the Examiner asserts that Mariani discloses a system that "makes a copy of the current object file," the Examiner does not specifically correlate a specific teaching in Mariani with either copying the object file, receiving a command to copy the object file, or copying the object file from a source project to a destination project. In the absence of such a correlation, we find that the Examiner has not demonstrated that Mariani discloses receiving a

command to copy a selected object from a source project to a destination project.

Accordingly, we conclude that Appellants have met their burden of showing that the Examiner erred in rejecting claims 6-9 and 15. Therefore, we reverse the Examiner's rejection of independent claims 6-9 and 15.

Issue #8

Did Appellants demonstrate that the Examiner erred in determining that the cited references disclose or suggest metadata?

Almond discloses a "meta model 300" (col. 6, l. 42) that "allows the system to map objects to representations . . ." (col. 6, l. 44). Because metadata includes any "data that provides information about other data" (Merriam-Webster's Collegiate Dictionary, 11th Edition, 2005), and because the meta model of Almond includes information that describes other information, we agree with the Examiner that Almond discloses or suggests metadata. For example, the meta model of Almond may "accommodate multiple names pointing to a single storage location" (col. 6, ll. 55-56). In this example of Almond, the multiple names point to (or provide location information about) a storage location of data. The multiple names of Almond constitute metadata because the multiple names provide information about other data.

For at least the aforementioned reasons, we conclude that Appellants have not sustained the requisite burden on appeal in providing arguments or evidence persuasive of error in the Examiner's rejection of claim 2 with respect to issue #8.

CONCLUSIONS OF LAW

Based on the findings of facts and analysis above, we conclude that Appellants have failed to demonstrate that the Examiner erred in:

1. finding that Mariani discloses or suggests determining a manner of executing a selected function on a selected object (issue #1),

2. finding that Fontana discloses automatically identifying dependent objects, causing functions to be performed, and executing a selected function on a selected object in an appropriate manner (issue #2),

3. combining Mariani with Fontana (issue #3),

4. determining that the recitation of "a reporting system" is given no patentable weight (issue #4),

5. finding that the Mariani reference discloses that each object has a unique identifier and a version identifier and manipulating objects between projects (issue #5),

6. finding that the Mariani reference discloses projects that reside in various environments (issue #6), and

7. determining that the cited references disclose or suggest metadata (issue #8).

However, Appellants have demonstrated that the Examiner erred in finding that the Mariani reference discloses receiving a command to copy a selected object from a source project to a destination project (issue #7).

DECISION

We affirm the Examiner's decision rejecting claims 1-5, 10-14, and 16-18 under 35 U.S.C. § 103. We reverse the Examiner's decision rejecting claims 6-9 and 15 under 35 U.S.C. § 103.

No time period for taking any subsequent action in connection with this appeal may be extended under 37 C.F.R. § 1.136(a)(1)(iv).

AFFIRMED-IN-PART

rwk

HUNTON & WILLIAMS LLP
INTELLECTUAL PROPERTY DEPARTMENT
1900 K STREET, N.W.
SUITE 1200
WASHINGTON, DC 20006-1109